Nekonvenční číselné systémy

Nekonvenční číselné systémy

- Nejrozšířenější číselný systém používaný v ALU:
  - Binární systém s dvojkovým doplňkem pro zobrazení záporných čísel
- Další číselné systémy se používají hlavně pro určité speciální aplikace:
  - Logaritmický číselný systém se znaménkem (LNS)
  - Kódy zbytkových tříd (RNS)
  - Číselný systém se záporným základem
  - Číselný systém se znaménky pro jednotlivé číslice

Logaritmický číselný systém (LNS)

= Logarithmic number system (LNS)

- pro repezentaci reálných čísel v poč.systémech (zejména zprac.sig.)
- v LNS je číslo X reprezentováno logaritmem své abs.hodnoty
  X → sign, log|X| + spec.reprezentace 0

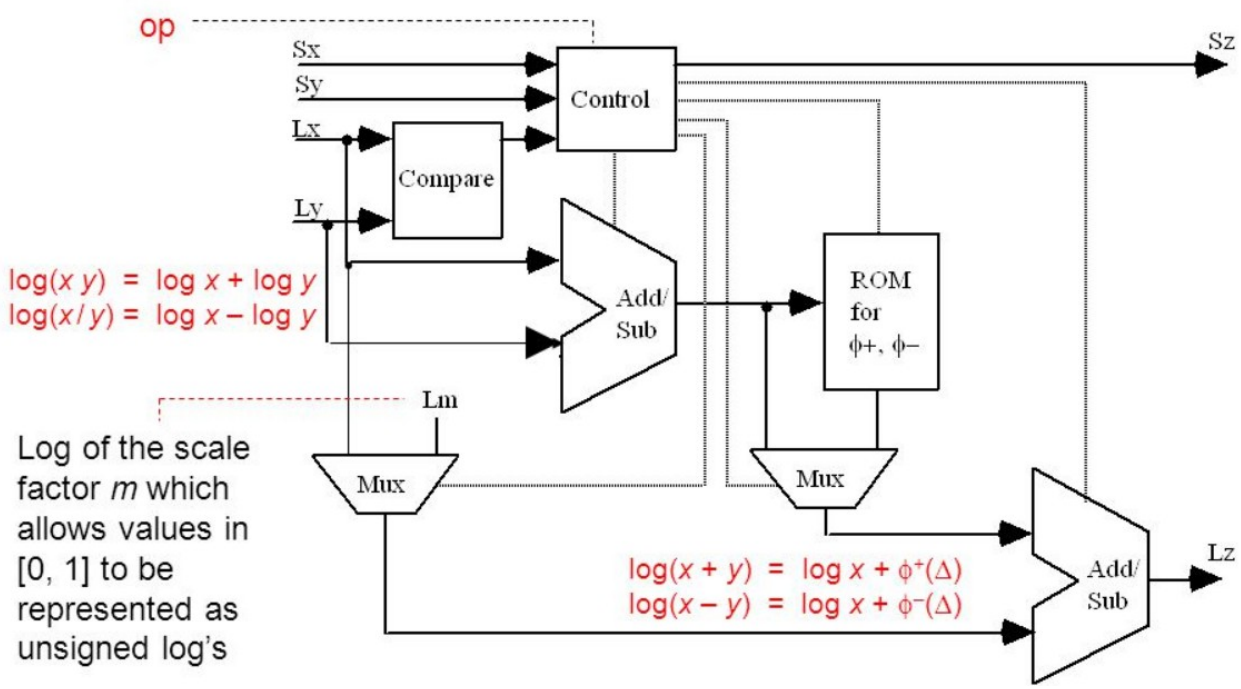
(rozsah reprezentace ve float-point pro X a fixed point LNS(X) je podobný)

- Operace v LNS (platí pro A>B):
  MUL: log(A*B) = log(A)+log(B)
  DIV: log(A/B) = log(A)-log(B)
  ADD: log(A+B) = log(A)+log(1+2^(log(B)-log(A)))
  SUB: log(A-B) = log(A)+log(1-2^(log(B)-log(A)))
  SQRT: log(sqrt(A)) = ½log(A)

..tedy zjednodušení u nasobení, dělení, mocnin, odmocnin,
..složitejší vyhodnocování plus a minus

Logaritmický číselný systém (LNS)

Struktura ALU (+,-,*,/) pro LNS
Pro +/- je nutné vyhodnotit fci 1 +/- 2^(Ly-Lx)



LNS se reálně/seriózně používá, může nýt výhodnější než klasický přístup (spotřeba, plocha, rychlost)

Kódy zbytkových tříd (RNS)

(Residue Number System - RNS)

- Modulární násobení je základní operací kryptografie
- Některé kryptografické metody vyžadují navíc modulární exponenciální funkci
- Kryptografické klíče jsou velmi dlouhá čísla (tisíce bitů)
- Otázka:

  Která reprezentace čísel je nejefektivnější?

Kódy zbytkových tříd

- Základy již ve 3. stol. n.l.
  - Sun Tzu: kniha Suan-ching

    We have things of which we do not know the number,
    If we count them by threes, the remainder is 2,
    If we count them by fives, the remainder is 3,
    If we count them by sevens, the remainder is 2,
    How many things are there?
    (kniha obsahuje metodu, jak nalézt odpověď = 23)
- 19. stol.
  - Carl Friedrich Gauss: *Disquisitiones Arithmeticae*
- Rok 1932 – první „vážný“ pokus o využití
  - D. H. Lenhmet: Special-purpose machine „photo-electric sieve“
    (určeno pro faktorizaci Mersennových čísel)

Kódy zbytkových tříd

- Nechť $m_1, m_2, \ldots, m_{k-1}, m_k$ (moduli) jsou přirozená navzájem nesoudělná (bez společného dělitele) čísla a $M = m_1 * m_2 * \ldots m_k$

Potom libovolné $X \in [0, M-1]$ můžeme reprezentovat pomocí k-tice čísel $(x_1, x_2, \ldots, x_k)$, kde:

$x_0 = X \bmod m_0$

$x_1 = X \bmod m_1$

...

$x_{k-1} = X \bmod m_{k-1}$

Z matematického hlediska se jedná o soustavu lineárních kongruencí, hledáme-li X.

- Každá taková k-tice $(x_0, x_1, \ldots, x_{k-1})$, kde $x_i \in [0, m_i-1]$ reprezentuje X jednoznačně v intervalu $X \in [0, M-1]$

$(m_0, m_1, \ldots, m_{k-1})$ tvoří bázi RNS a označujeme ji B_k.

Kódy zbytkových tříd

Označení: $X=(x_0|x_1|...|x_{k-1})RNS(m_0|m_1|...|m_{k-1})$

Jinak: $x_i = X \bmod m_i$

$m_i>m_j$ pro všechny i<j .. moduly jsou seřazené

Příklady:

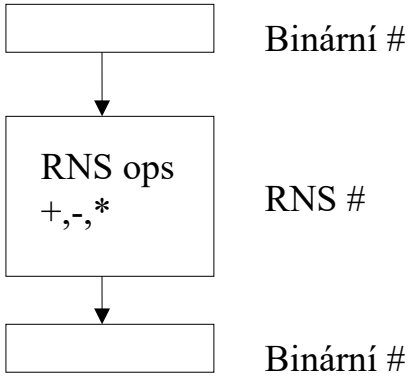
84 = (0|4|0)RNS(7|5|3)
1 = (1|1|1)RNS(7|5|3)
2 = (2|2|2)RNS(7|5|3)
3 = (3|3|0)RNS(7|5|3)
123 = (3|4|3|0)RNS(8|7|5|3)

Residuální číselný systém není poziční.

Pro moduly (7|5|3) existuje jednoznačná reprezentace čísel rozsahu 0-104.

Kódy zbytkových tříd

- Potřeba obousměrné konverze



- Jaké výhody přinášejí operace v kódech zbytkových tříd RNS?

Zavedeme další označení: $X \bmod m = \langle X \rangle_m$

Kódy zbytkových tříd

Matematické operace v RNS:

- Sčítání (odčítání je podobné) - součet Xi+Yi modulo Mi

$$x+y = (\langle x_0+y_0 \rangle_{m_0} | \langle x_1+y_1 \rangle_{m_1} | \ldots | \langle x_{k-1}+y_{k-1} \rangle_{m_{k-1}})RNS(m_0|m_1|\ldots|m_{k-1})$$

kde $x_i = \langle x \rangle_{m_i}$ a $y_i = \langle y \rangle_{m_i}$

- Násobení – vynásobení Xi*Yi modulo Mi

$$x*y = (\langle x_0*y_0 \rangle_{m_0} | \langle x_1*y_1 \rangle_{m_1} | \ldots | \langle x_{k-1}*y_{k-1} \rangle_{m_{k-1}})RNS(m_0|m_1|\ldots|m_{k-1})$$

kde $x_i = \langle x \rangle_{m_i}$ a $y_i = \langle y \rangle_{m_i}$

- Celočíselné mocniny – mocnina Xi^P modulo Mi
- Dělení ? - Obtížné.

Kódy zbytkových tříd

Záporná čísla v RNS:

- Možno reprezentovat doplňkem (doplňek M)
- Zbytky pro -X jsou stejné jako zbytky M-X

  $-x \bmod m_i = (M-x) \bmod m_i$

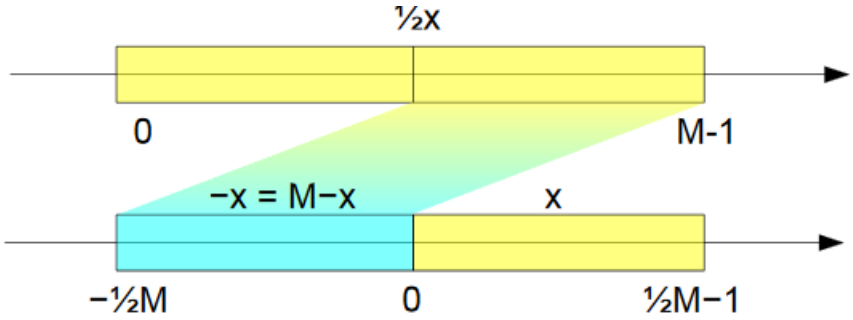
  $\rightarrow$ zbytky -x jsou m_i-zbytek x

  $x = (x_{k-1}| \,..\, |x_0) \rightarrow -x = (m_{k-1}-x_{k-1}| \,..\, |m_0-x_0)$

Např.

x=21=(5|0|1|0)RNS(8|7|5|3)

-x=-21= (8-5|7-0|5-1|3-0) = (3|0|4|0) RNS(8|7|5|3)



Kódy zbytkových tříd

Výběr modulů (příklad) pro M=100000(=17bit)

jednotivé zbytky jsou reprezentovány jako samostatná binární čísla, např. 5 = (5|5|0|2)RNS(8|7|5|3)→ 101 101 000 10 (11-bits)

účinnost reprezentace je poměr rozsahu RNS s mořným rozsahem binárního ekvivalentu, např. 8*7*5*3 / 2^11 = 41%

- Po sobě jdoucí prvočísla

  RNS(17|13|11|7|5|3|2) → M=510510, 23 bits
- Odstranění prvočísla (předchozí zbytečně velký rozsah)

  RNS(17|13|11|7|3|2) → M=102102, 20 bits
- Kombinování prvočísel – úprava délky/počtu modulů

  RNS(26|21|17|11) → M=102102, 19 bits
- Užití malých (prvo)čísel, užití neprvočísla

  RNS(15|13|11|2^3|7) → M=120120, 18 bits, max 4-bit field
- výběr 2^n a 2^n-1 modulů

  RNS(2^5|2^5-1|2^4-1|2^3-1) → ∏=104160, 17 bits, eff ≈ 100%

Věta: 2^a-1 & 2^b-1 jsou nesoudělná tehdy a jen tehdy, jestliže a & b jsou nesoudělná

Kódy zbytkových tříd

- Výhody:
  - Paralelní výpočet +,*,- který navíc operuje jen s malými čísly
  - Přidáváním prvočísel bez zvyšování rozsahu lze zvyšovat oddolnost proti poruchám (fault tolerance) (Př: přechod od (7|5|3) na (7|5|3|2), kde poslední pole je použito pro paritu)
- Nevýhody:
  - Nelze jednoduše provádět operaci dělení
  - Komparace není triviální
  - Poměrně náročná konverze do binární formy
  - Obtížná detekce přetečení
- Jak provést konverzi?

Kódy zbytkových tříd

- **Konverze binární (NBC) -> RNS**

  Použití tabulek (lookup table = LUT) - odstraňuje nutnost dělení velkých čísel (“velké“ číslo v NBC/m_i) - Číslo v NBC rozložíme na součet binárních řádů použijeme LUT pak zbyde jen jednoduché dělení m_i.

$$b_{k-1}...b_0 \bmod m_i = (2^{k-1}b_{k-1} \bmod m_i + \ldots + 2^0 b_0 \bmod m_0) \bmod m_i$$

NBC number — i-th residue of RNS representation

Precomputed for given RNS — Precomputed for given RNS

requires division of relatively small number - can be implemented with LUT

Příklad: 164=10100100 → RNS(8|7|5|3)
(k=číslo bitu, počítáme jen jedničkové)
(tedy 2^7, 2^5, 2^2 mod mi – viz LUT vpravo)
10100100 mod 8 = (0+0+4) mod 8 = 4
10100100 mod 7 = (2+4+4) mod 7 = 3
10100100 mod 5 = (3+2+4) mod 5 = 4
10100100 mod 3 = (2+2+1) mod 3 = 2
→ (4|3|4|2)RNS(8|7|5|3)

		2^k mod …			
k	2^k	8	7	5	3
0	1	1	1	1	1
1	2	2	2	2	2
2	4	4	4	4	1
3	8	0	1	3	2
4	16	0	2	1	1
5	32	0	4	2	2
6	64	0	1	4	1
7	128	0	2	3	2

Kódy zbytkových tříd

Konverze binární (NBC) -> RNS

Dáno $(x_0|x_1|...|x_{k-1})RNS(m_0|m_1|\dots|m_{k-1})$

Binární číslo X = SUM($<x_i*w_i>_{mi}$ * M/m_i)

Předem vypočítané konstanty

Viz: Čínská věta

$$\left(\sum_{0}^{k-1}\left(Xi*Wi\right)_{mi}*\frac{M}{mi}\right)_{M}$$

kde $w_i = inv\left(\frac{M}{m_i}\right) \Leftrightarrow \left(w_i*\frac{M}{m_i}\right)_{m_i} = 1$ $\qquad M=\prod_{i=0}^{k-1} m_i$

Konstanty w_i (inverzní prvky) jsou pro daný RNS vypočítány jen jednou provždy a tvoří jeho nedílnou součást

Kódy zbytkových tříd

- Příklad konverze RNS -> binární (NBC = normal bin.code)

  (6|4|0)RNS(7|5|3) … převod čísla (6|4|0) v RNS(7|5|3)

Pro RNS (7|5|3) nejprve najdeme (předpočítáme) w_i

$\langle w_1 * 5 * 3 \rangle_7 = 1 \Rightarrow w_1 : 1$ (tj. Hledáme w1 takové že platí w1*5*3 mod 7 = 1)

$\langle w_2 * 7 * 3 \rangle_5 = 1 \Rightarrow w_2 : 1$

$\langle w_3 * 7 * 5 \rangle_3 = 1 \Rightarrow w_3 : 2$

Pak můžeme algorimicky převádět lib.číslo (v našem případě (6|4|0)):

$X = \langle 6 * w_1 * 5 * 3 * 7/7 + 4 * w_2 * 7 * 3 * 5/5 + 0 * w_3 * 7 * 5 * 3/3 \rangle_{105}$

$X = \langle 6 * w_1 * 15 + 4 * w_2 * 21 + 0 * w_3 * 35 \rangle_{105}$

$X = \langle 6 * 1 * 15 + 4*1*21 + 0*2*35 \rangle_{105} = \langle 174 \rangle_{105} = 69$

ověření: (69 mod 7=6, 69 mod 5=4, 69 mod 3=0)

Kódy zbytkových tříd

- Příklad konverze RNS -> binární #:

1 = (1|1|1)RNS(7|5|3)

$\langle w_1 * 5 * 3 \rangle_7 = 1 \Rightarrow w_1 : 1$

$\langle w_2 * 7 * 3 \rangle_5 = 1 \Rightarrow w_2 : 1$

$\langle w_3 * 7 * 5 \rangle_3 = 1 \Rightarrow w_3 : 2$

$X = \langle 1 * w_1 * 5 * 3 * 7/7 + 1 * w_2 * 7 * 3 * 5/5 + 1 * w_3 * 7 * 5 * 3/3 \rangle_{105}$

$X = \langle 1 * w_1 * 15 + 1 * w_2 * 21 + 1 * w_3 * 35 \rangle_{105}$

$X = \langle 1 * 1 * 15+1 * 1 * 21+1 * 2 * 35 \rangle_{105} = \langle 106 \rangle_{105} = 1$

Kódy zbytkových tříd

- Porovnání (komparace)
  - Je možná konverze RNS -> bin a pak lze provést standardní porovnání.
  - Jiný přístup: převod do systému se smíšeným základem (MRS – mixed radix system), ve kterém lze čísla porovnat.
  - Pro provádění aritmetických operací není systém MRS příliš výhodný, porovnání velikosti dvou čísel lze ale provést jednoduše.

Systém se smíšeným základem (MRS)

Definujme systém se smíšeným základem následovně:

MRS (mixed radix system):

$X = (Z_{k-1}|Z_{k-2}|...Z_0)MRS(m_{k-1}|m_{k-2}|...|m_0)$

$X = Z_{k-1}m_{k-1}m_{k-2}...m_1 + Z_{k-2}m_{k-2}m_{k-1}...m_1 + ... + Z_2m_2m_1 + Z_1m_1 + Z_0$

Pro účel porovnání použijeme stejné moduly m_i jako pro RNS !

Konverze z RNS do MRS:

Dáno $X = (x_{k-1}|x_{k-2}|...|x_0)RNS(m_{k-1}|m_{k-2}|...|m_0)$

Chceme určit $X = (Z_{k-1}|Z_{k-2}|...|Z_0)MRS(m_{k-1}|m_{k-2}|m_{k-3}|...|m_0)$

Vlastnost:

- Cifra MRS Z_0 je pouze v řádu jednotek, takže pro MRS potřebujeme méně prvočísel než pro RNS

Systém se smíšeným základem

- Otázka: jaký je vztah mezi x_0 a Z_0?

Lze jej jednoduše nalézt? Odpověď - ano.

1 $x_0 = Z_0$.

- Proč? x_0 je zbytek po dělení čísla x mod m_0 – všechny ostatní části čísla jsou násobkem m_0 -> $x_0 = Z_0$

Z toho vyplývá

2 $x-x_0 = (x'_{k-1}|x'_{k-2}|...|x'_1|-)RNS(m_{k-1}|m_{k-2}|...|m_1|-)$
$= (Z_{k-1}|Z_{k-2}|...|Z_1|0)MRS(m_{k-2}|m_{k-3}|...|m_0)$

kde $x'_i = \langle x_i - x_0 \rangle_{m_i}$

Systém se smíšeným základem

- To vede na vztah:

3 $(x-x_0)/m_0 = (x''_{k-1}|x''_{k-2}|...|x''_1|-)RNS(m_{k-1}|m_{k-2}|...|m_1|-)$
$= (Z_{k-1}|Z_{k-2}|...|Z_1)\ MRS\ (m_{k-2}|m_{k-1}|...|m_1)$

$Z_1 = x''_1$

a tak se pokračuje dále. (dedukce, dělení, opakování)

Přestože jsme na počátku tvrdili, že dělení je v RNS obtížné, nyní jsme prováděli operaci dělení.

Trik?

V tomto případě víme, že výsledek bude vždy číslo celé.

Zpět do RNS

- Dělení v RNS

$x''_i = x_i * \langle m_0^{-1} \rangle_{m_i}$

kde

$\langle m_j^{-1} \rangle_{m_i}$ je multiplikativní inverzní prvek m_j vzhledem k m_i

Příklad: $\langle 3^{-1} \rangle_7 = 5$ --> $\langle 3 * \langle 3^{-1} \rangle_7 \rangle_7 = 1$

$\langle 3^{-1} \rangle_5 = 2$ --> $\langle 3 * \langle 3^{-1} \rangle_5 \rangle_5 = 1$

Příklad: RNS -> MRS

$Y=(1|3|2)RNS(7|5|3) = (Z_2|Z_1|Z_0)MRS(5|3)$

To znamená, že $Y = Z_2*5*3+Z_1*3+Z_0$

podle 1

$Z_0 = x_0 = 2$

podle 2 dostaneme

$y-x_0 = y-2 =(x'_2|x'_1|0)RNS(7|5|3) = (Z_2|Z_1|0)MRS(5|3)$

$x'_2 = \langle x_2-x_0 \rangle_{m_2} = \langle 1-2 \rangle_7 = 6$

$x'_1 = \langle 3-2 \rangle_5 = 1$

$y-2 = (6|1|0)RNS(7|5|3)=(Z_2|Z_1|0)MRS(5|3)$

Příklad - pokračování

z výrazu 3 dostaneme

$(y-x_0)/m_0 = (y-2)/3 = (x''_2|x''_1|-)RNS(7|5|3) = (Z_2|Z_1)MRS(5)$

potom $$\left(\frac{7A + x_2^{'}}{3}\right)_7 = \left(7B + x_2^{''}\right)_7$$

Otázka: Jak se odvodí x''_2 ?

Odpověď: “Těžko.” Je třeba zkoušet čísla až do hodnoty modulu

$$\left(7A + x_2^{'}\right)_7 = \left(21B + 3x_2^{''}\right)_7$$

pamatujte, že $(3^{-1})_7 = 5$, $(3^{-1})_5 = 2$

$x''_2 = (5*6)_7 = 2$, $x''_1 = (1*2)_5 = 2$

$(y-2)/3 = (2|2|-)RNS(7|5|3) = (Z_2|Z_1)MRS(5)$

Příklad - pokračování

Aplikujeme znovu 1,

$Z_1 = x''_1 = 2$

dále podle 2

$(y-x_0)/m_0 - x''_1 = (x'''_2|0|-)RNS(7|5|3) = (Z_2|0)MRS(5)$

$x'''_2 = \langle x''_2 - 2)_7 = \langle 2-2)_7 = 0$

$(0|0|-)RNS(7|5|3) = (Z_0|0)MRS(5)$

Podle 3

$x''''_2 = \langle x'''_2 * \langle 5^{-1})_7)_7 = 0 = Z_2$

$$\frac{\frac{y - x_0}{P_0} - x''_1}{P_1} =$$

Dostaneme konečný výsledek

$(0|2|2)MRS(5|3)$

Příklad na závěr

Ověřte správnost

(0|2|2)MRS(5|3) = 0*5*3+2*3+2 = 8

$\langle 8 \rangle_7 = 1, \langle 8 \rangle_5 = 3, \langle 8 \rangle_3 = 2$ -> (1|3|2)RNS(7|5|3)

Platí!

Číselné systémy se záporným základem

- Základ r číselného systému s konstantním základem – obvykle kladné celé číslo r > 1
- Základ r může být i záporný: $r = -\beta$ (β - kladné celé číslo)
- Rozsah číslic: $x_i = 0,1,\ldots,\beta-1$
- Hodnota n-tice $(x_{n-1},x_{n-2},\ldots,x_0)$: $$X = \sum_{i=0}^{n-1} x_i (-\beta)^i$$
- Váha w_i je rovna: $$w_i = \begin{cases} \beta^i & \text{pro sudá i} \\ -\beta^i & \text{pro lichá i} \end{cases}$$

Příklad – negativní desítková soustava

- Dekadický číselný systém se záporným základem $\beta=10$
- Čísla negativního dekadického systému o šířce tří číslic:
  - $(192)_{-10} = 100-90+2=12$; $(012)_{-10} = -10+2=-8$
  - Největší kladné číslo $(909)_{-10} = (909)_{10}$
  - Nejmenší číslo $(090)_{-10}=(-90)_{10}$
  - Asymetrický obor hodnot $-90 \leq X \leq 909$
  - Přibližně 10 x více kladných než záporných čísel
- Toto platí pro lichá n, pro sudá n platí opak
- Příklad: pro $n = 4$ je rozsah roven $-9090 \leq X \leq 909$

Vlastnosti systému se záporným základem

- Není třeba vyhrazený znaménkový bit
- Zobrazení záporných čísel je „automatické“.
- Znaménko čísla je určeno první nenulovou číslicí čísla
- Mezi reprezentací kladných a záporných čísel není rozdíl, aritmetické operace jsou z hlediska znaménka indiferentní
- Algoritmy základních operací pro negativní základ jsou mírně složitější než v případě konvenčního číselného systému

Příklad binárního systému se záporným základem

Záporná báze $\beta = 2$ délka čísla $n = 4$

Rozsah zobrazení: $-(1010) = (1010)_{-2} \leq X \leq (0101)_{-2} = +(5)_{10}$

Při součtu čísel mohou být přenosy kladné nebo záporné:

- Příklad: -8 +4 -2 +1

0	0	1	0	-2
0	0	1	1	-1
1	1	0	1	-3

Předpokládané aplikace – zpracování signálů
Algoritmy existují pro všechny aritmetické operace

Nestal se populárním.
Hlavní důvod: Nevychází lépe než dvojkový doplněk

Číselné systémy s ciframi se znaménkem

Pozn. Podtržení = negace

Varianty:

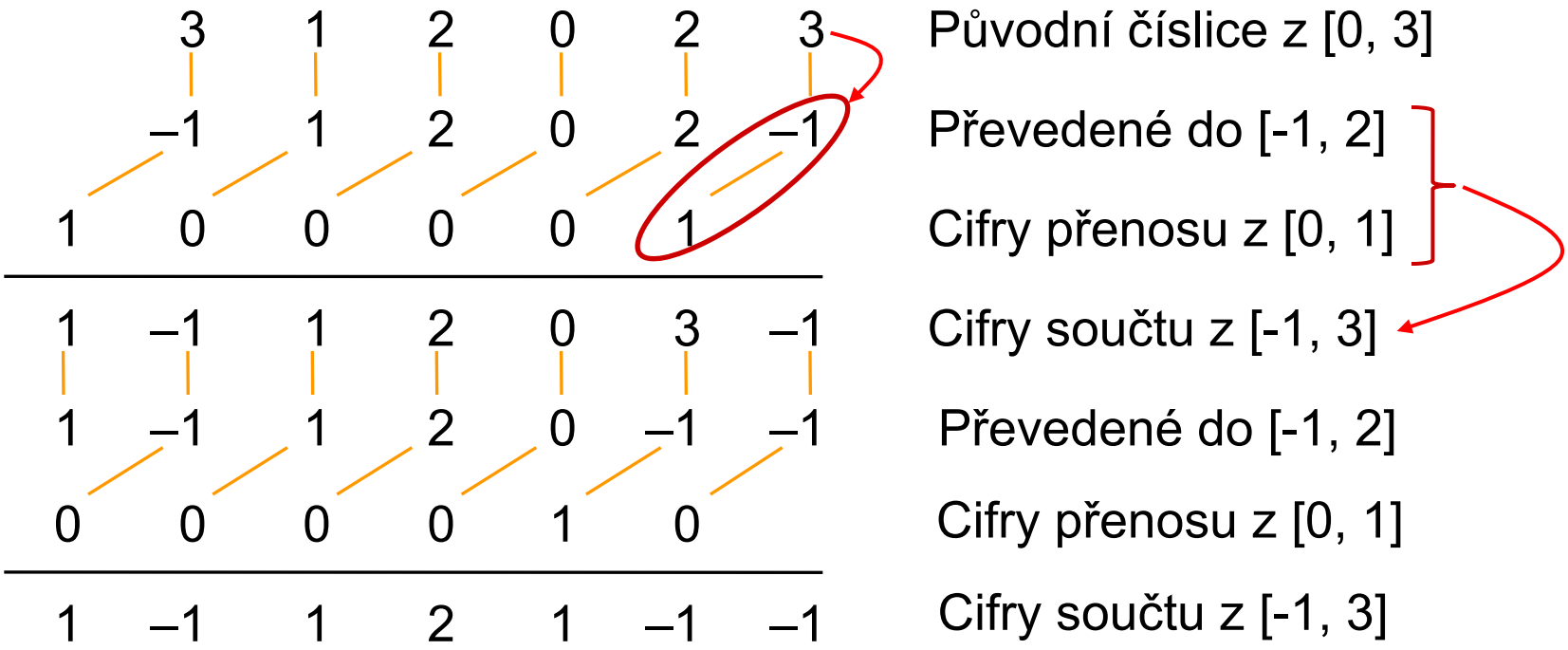
- Redundantní číselné systémy s oborem cifer symetrickým kolem nuly.
  - Rozsah číslic: $x_i \in \{\underline{r-1}, \underline{r-2}, \ldots, \underline{1}, 0, 1, \ldots, r-1\}$
- Redundantní číselné systémy s nesymetrickým rozložením cifer kolem nuly.
  - Rozsah číslic: $x_i \in \{-\alpha, \underline{1}\ 0, 1, \ldots, \beta\}$, kde $\alpha + \beta + 1 - r > 0$
- Hodnota n-tice $(x_{n-1}, x_{n-2}, \ldots, x_0)$ je v obou případech rovna:

$$X = \sum_{i=0}^{n-1} x_i r^i$$

Znaménko pro každou cifru

Reprezentace cifer se znaménkem: Množina cifer [$-\alpha$, β] namísto [0, $r-1$]

Příklad: Reprezentace základ 4 s ciframi [–1, 2] namísto [0, 3]

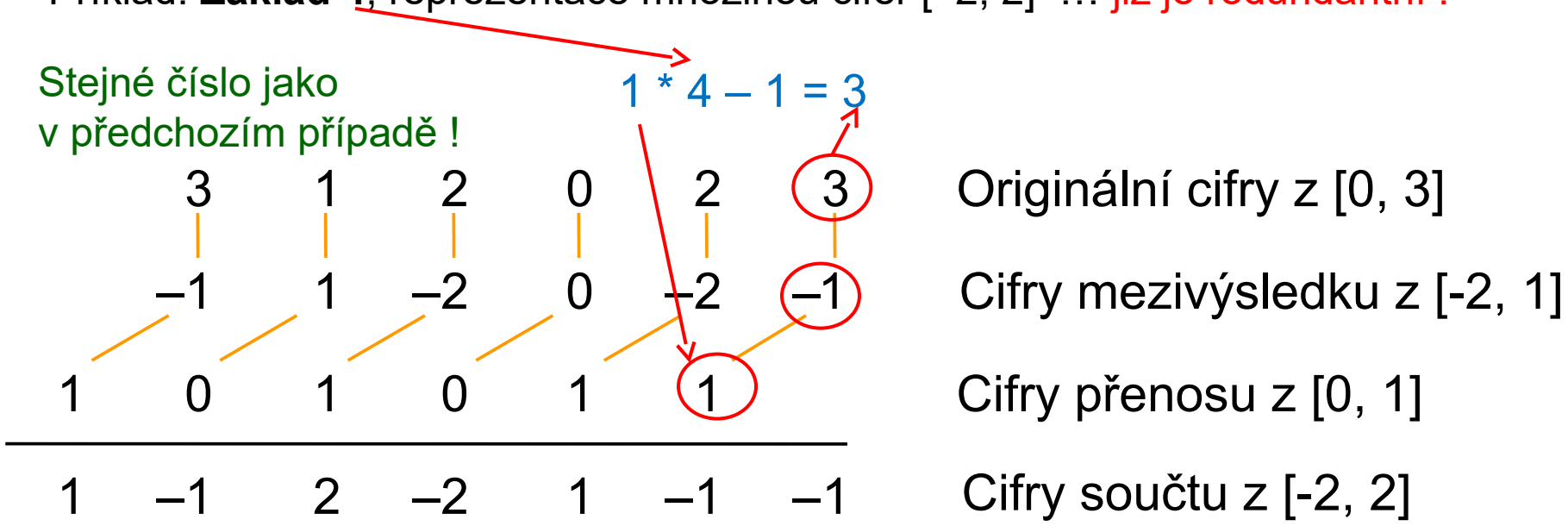


Převod standardního vyjádření integer se základem 4 do nestandardního formátu integer s ciframi z [–1, 2].

Redundantní reprezentace s ciframi se znaménkem

Reprezentace cifer se znaménkem: Množina cifer $[-\alpha, \beta]$, kde $\rho = \alpha + \beta + 1 - r > 0$

Příklad: **Základ 4**, reprezentace množinou cifer [–2, 2] … již je redundantní !



Konverze standardního vyjádření integer se základem 4 na integer se základem 4 a nestandardní množinou cifer [–2, 2].

V tomto případě přenos neprochází dále, konverze je “carry-free”.

Číselné systémy s redundancí

- Zvažte výhody a nevýhody redundantního vyjádření čísel

- Redundance eliminuje dlouhé přenosové řetězce (carry)
- Redundance může mít mnoho forem
- Konverze mezi redundantní a neredundantní reprezentací?
- Má se redundantní vyjádření použít i pro konečnou formu výsledku?

Problémy s přenosem

Možnosti jak se vypořádat s dlouhou přenosovou cestou:

1. Omezit délku přenosu na malý počet bitů
2. Detekovat ukončení přenosu (nečekat na nejhorší případ) – asynchronní sčítačka – již znáte
3. Zrychlení průchodu přenosu (Carry Look Ahead, atd.)
4. Ideální: Úplně vyloučit přenosy

```
   5   7   8   2   4   9
+  6   2   9   3   8   9   Cifry operandů z [0, 9]
-------------------------
  11   9  17   5  12  18   Součty cifer z [0, 18]
```

Lze to rozšířit za hranice jednoduchého součtu?

Cifry výsledku jsou sice v rozsahu [0, 18], ale přenosy se při sčítání nekonaly !!!

Sčítání čísel s redundancí

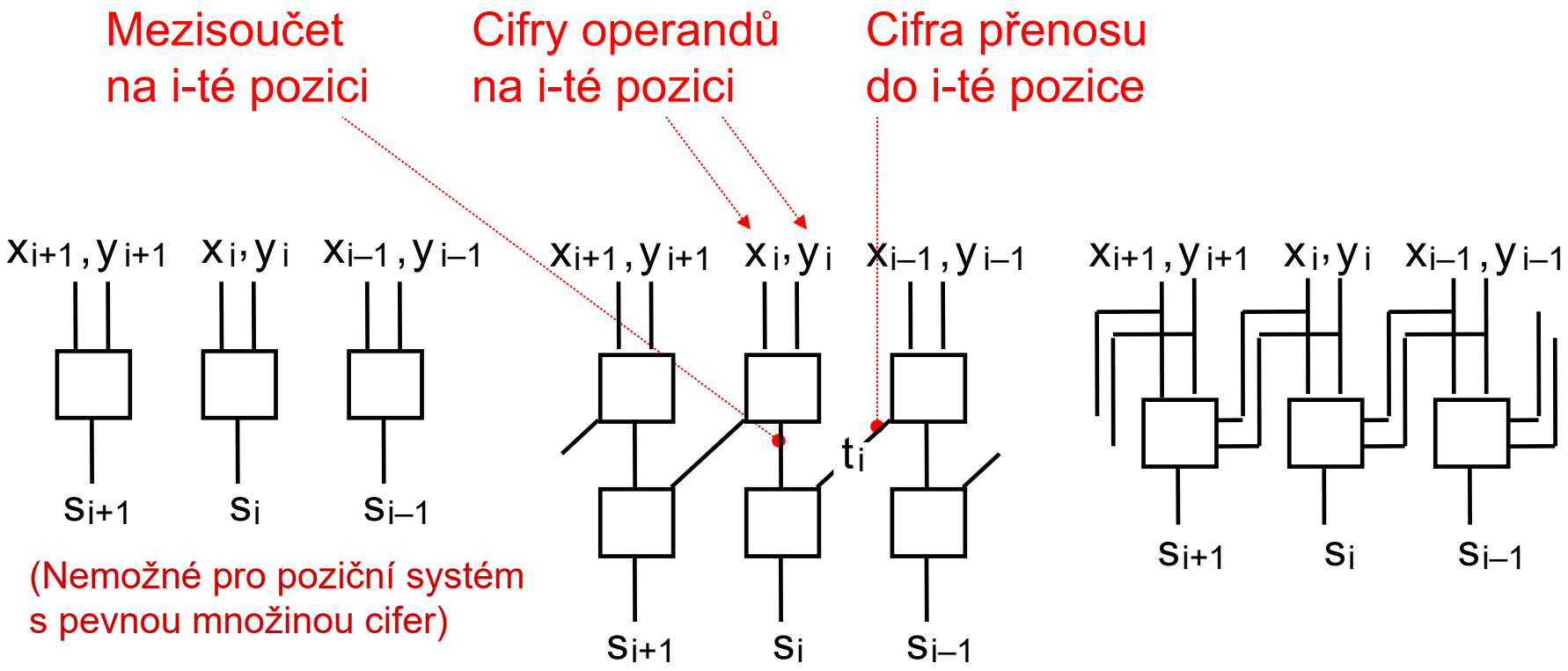
Dekompozice dílčích součtů $[0, 36] = 10 \times [0, 2] + [0, 16]$

Absorpce cifer přenosu $[0, 16] + [0, 2] = [0, 18]$

	11	9	17	10	12	18	Cifry operandů z [0, 18]
+	6	12	9	10	8	18	
	17	21	26	20	20	36	Dílčí součty z [0, 36]
	7	11	16	0	10	16	Mezisoučty z [0, 16]
1	1	1	2	1	2		Cifry přenosu z [0, 2]
1	8	12	18	1	12	16	Cifry součtu z [0, 18]

Sčítání desítkových čísel v množině cifer [0, 18].

Sčítání „bez přenosu“



(a) Ideální jednostupňové sčítání bez přenosu.

(b) Dvojstupňové sčítání bez přenosu.

(c) Jednostupňové sčítání s „lookahead“.

Ideální a reálná schémata „carry-free“ sčítání.

Index redundance

Redundancí dosahujeme „bezpřenosového“ sčítání $-\alpha$ β

Jaká míra redundance je třeba? Je dostačující [0, 11] pro $r = 10$?

Index redundance $\rho = \alpha + \beta + 1 - r$ Například, $0 + 11 + 1 - 10 = 2$

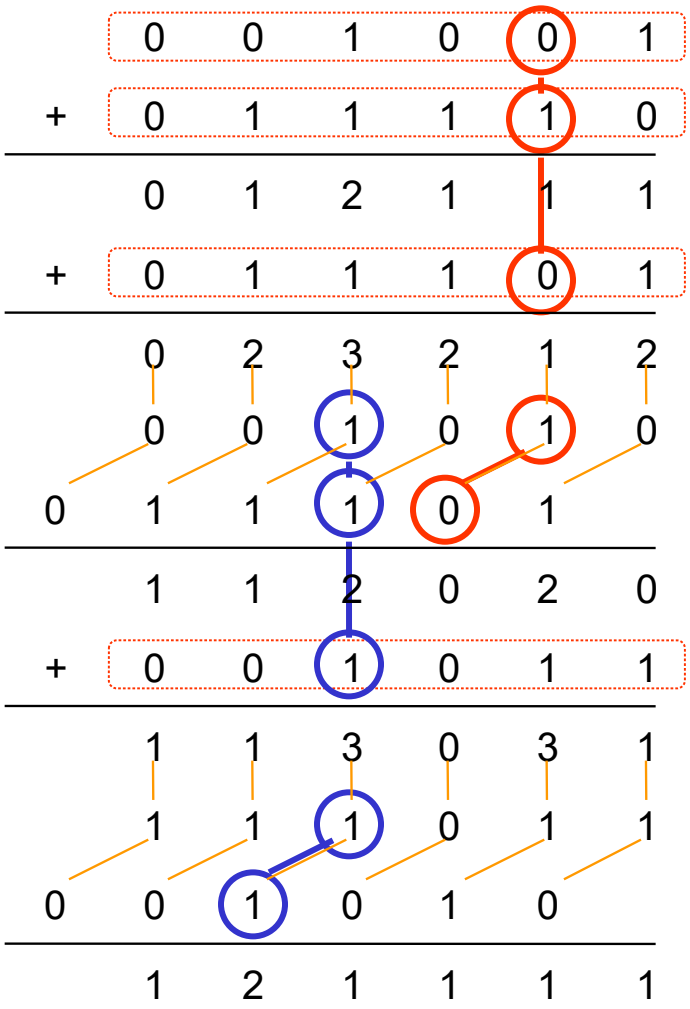
	11	10	7	11	3	8	
+	7	2	9	10	9	8	Cifry operandů v [0, 11]
	18	12	16	21	12	16	Ciferné součty v [0, 22]
	8	2	6	1	2	6	Mezisoučty v [0, 9]
1	1	1	2	1	1		Cifry přenosu v [0, 2]
1	9	3	8	2	3	6	Cifry součtu v [0, 11]

Sčítání čísel o základu 10 v množině cifer [0, 11].

Redundance v počítačové aritmetice

Nejstarší příklad redundance v počítačové aritmetice je reprezentace uloženého přenosu (carry-save addition)

Součet čtyř dvojkových čísel, kdy suma je vyjádřena s uchovaným přenosem (stored-carry form).



Systémy – číslice se znaménkem

- Dosud jsme uvažovali soubor číslic {0,...,r-1}
- V systému s ciframi se znaménkem (SD) mohou číslice nabývat hodnot v rozsahu {r-1,r-2,…,1,0,1,...,r-1} (1 = -1)
- Není třeba používat vyhrazené znaménko
- Příklad:
  - r = 10, n = 2 ; cifry - {9,8,...,1,0,1,...,8,9}
  - Rozsah: $99 \leq X \leq 99$ to je celkem 199 čísel
  - 2 cifry, 19 možností každá - 361 reprezentací => redundance
  - 01=19=1 ; 02=18 = -2
  - Reprezentace 0 (nebo 10) je jednoznačná
  - Z celkového počtu 361 reprezentací, 361-199 = 162 je redundantních - redundance je 81%
  - Každé číslo má maximálně dvě reprezentace

Vyjádření redundance také v %